

## WHAT IS CLAIMED IS:

1. A method for interfacing between a multi-threaded application and a restrictive back-end processing system, wherein the back-end processing system requires a thread-dependent connection where a relationship between a connection and an application thread is maintained, the method comprising:

detecting a thread-dependent connection in the back-end processing system;

maintaining a single thread to link to the detected thread-dependent connection;

correlating multiple threads from the application with the maintained single thread, thereby allowing operations requested by the application over the multiple threads to be performed on the back-end processing system through the thread-dependent connection.

2. The method of claim 1, wherein detecting a thread-dependent connection comprises detecting the application attempting to access the back-end processing system.

3. The method of claim 1, wherein the connector allocates memory for connection instances, and wherein detecting a thread-dependent connection comprises detecting the connector allocating memory for a connection instance.

4. The method of claim 1, wherein detecting a thread-dependent connection comprises reading a header data structure for the connector, the header data structure storing identification data including data identifying the thread requirements of the connector.

5. The method of claim 4, wherein reading a header data structure for the connector comprises reading a flag in the header data structure, the flag indicating whether the connector support multiple threads per connection.

6. The method of claim 1, wherein maintaining a single thread comprises receiving a thread from the application and isolating the single thread as the single thread to link to the connection.

7. The method of claim 1, wherein the thread-dependent connector is configured to generate a plurality of simultaneous connections, comprising:

detecting a second thread-dependent connection in the back-end processing system;

maintaining a second single thread to link to the detected second thread-dependent connection; and

correlating one or more second threads from the application with the maintained second single thread, thereby allowing additional operations requested by the application over the second thread(s) to be performed on the back-end processing system through the second thread-dependent connection.

8. The method of claim 1, wherein correlating multiple threads from the application comprises mapping each of the multiple threads with the maintained single thread.

9. The method of claim 1, wherein correlating multiple threads from the application comprises generating a thread support object to support relationships between the multiple threads and the maintained single thread and using the thread support object to toggle execution of an operation between one of the multiple threads and the maintained single thread.

10. The method of claim 9, wherein the thread support object comprises two or more semaphores, and comprising using the two or more semaphores to toggle execution of the operation between one of the multiple threads and the maintained single thread.

11. A thread consistency support system for providing thread consistency between a multi-threaded application and a thread-dependent connector allocated in a restrictive back-end system, wherein the thread-dependent connector only supports a single thread to link to that connector for operations on that connector, and wherein the multi-threaded application creates multiple threads that attempt to access the connector, the system comprising:

an arbiter layer positioned between the application and the thread-dependent connector, the arbiter layer being configured to receive multiple threads from the application and to produce a single internal thread from the arbiter layer to the connector upon which operations of the multiple threads are performed.

12. The system of claim 11, further comprising an activation detector that activates the arbiter layer in response to the activation detector detecting a multi-threaded application attempting to access the restrictive back-end system.

13. The system of claim 12, wherein the activation detector is an enhancement incorporated into a connector application program interface.

14. The system of claim 11, wherein the arbiter layer channels the multiple threads from the application through a single internal thread by mapping the threads to preserve one thread per thread-dependent connector.

15. The system of claim 11, comprising a thread isolation routine for isolating a thread from the multiple threads of the application to a single internal thread for linking to a thread-dependent connection.

16. The system of claim 11, comprising a toggle routine channels the multiple threads from the application to produce a single internal thread by using threading in connection with to isolate thread execution.

17. The system of claim 11, wherein the single internal thread produced by the arbiter layer acts as a thread sub-connection to the thread-dependent connector that is assigned to each original multiple thread connection from the application to the arbiter layer.

18. The system of claim 17, wherein the thread consistency support system utilizes connector methods, and wherein the connector methods appear to the application to be those of the underlying sub-connected thread-dependent connector.

19. The system of claim 11, wherein the system establishes a connection handle for the single internal thread with the thread-dependent connector that is returned to the connector application program interface of a calling application, and wherein the thread consistency support system utilizes the connection handle to identify and employ the internal thread when interacting with the thread-dependent connector in response to requests from the multi-threaded application.

20. The system of claim 11, wherein the system is configured to channel multiple threads from a multi-threaded application to more than one thread-dependent connector.

21. A thread consistency support system for providing thread consistency from a connector application program interface that creates multiple threads to a thread-dependent connector that only allows a single thread to link to that connector for all operations on that connector, the system comprising:

a threading meta-connector interacting between the connector application program interface and the thread-dependent connector, wherein the threading meta-connector establishes a connection handle for a single internal thread with the thread-dependent connector that is returned to the connector application program interface of a calling multi-threaded application in place of connection handles requested for multiple application threads, in response to the

threading meta-connector's receipt of multiple application threads from the connector application program interface; and

an activation detector that activates the threading meta-connector in response to the activation detector identifying a multi-threaded application attempting to access a thread-dependent connector;

thereby ensuring that the single internal thread that initializes a connection from the thread-dependent connector is used for all subsequent operations attempted by the multiple application threads from the multi-threaded application to that thread-dependent connector.

22. A method of providing thread consistency from a connector application program interface that creates multiple threads to a thread-dependent connector that only allows a single thread to link to that connector for all operations on that connector, the method comprising:

receiving multiple application threads from the connector application program interface;

creating a single internal thread that links with the thread-dependent connector;

and

performing data operations of the multiple application threads from the connector application program interface over the single internal thread link with the thread-dependent connector.